BIT manuscript No.
(will be inserted by the editor)

Low cardinality Positive Interior cubature on NURBS-shaped domains

Alvise Sommariva · **Marco Vianello**

Received: date / Accepted: date

Abstract In this paper we propose an algorithm that computes a low cardinality PI-type (Positive weights and Interior nodes) algebraic cubature rule of degree n, with at most $(n+1)(n+2)/2$ nodes, over curvilinear polygons defined by piecewise rational functions. Typical examples are domains whose boundary is defined piecewise by NURBS curves or by composite Bezier curves. The key tools are a relevant but overlooked theorem of 1976 by Wilhelmsen on *Tchakaloff sets*, a specific *in-domain* algorithm for such curvilinear polygons and the sparse nonnegative solution of underdetermined moment matching systems by the Lawson-Hanson NonNegative Least Squares solver. Many numerical tests are performed to show the flexibility of this approach and the implemented MATLAB toolbox is freely available to the users, in particular for possible applications within FEM/VEM with NURBS-shaped curvilinear elements.

Keywords low-cardinality algebraic cubature · PI (Positive Interior) rules · curvilinear polygons, NURBS · crossing number · winding number · Tchakaloff sets · underdetermined moment matching systems · NonNegative Least Squares · NEFEM (NURBS-Enhanced FEM) · VEM (Virtual Element Method).

Mathematics Subject Classification (2000) 65D07 · 65D10 · 65D17 · 65D18 · 65D32 · 65N30.

1 Introduction

The need of efficient algebraic cubature formulas (that are cubature formulas with a given degree of polynomial exactness) on curvilinear polygonal elements (that are polygons with one or more curved sides), is not new but has manifestly re-emerged during the last fifteen years in the framework of the numerical solution of PDEs by

Alvise Sommariva, Marco Vianello
University of Padova, Italy
E-mail: alvise@math.unipd.it,marcov@math.unipd.it

FEM and VEM approaches. Such curvilinear elements for example typically arise when a polygonal mesh intersects a curved domain boundary and allow to cope accuracy loss due to standard piecewise linear approximation of the boundary itself, but the relevant algebraic cubature problem can become a bottleneck of the implementation if not properly treated. In particular, the case of curvilinear polygons with NURBS sides is at the base of the NEFEM (NURBS-Enhanced Finite Element Method) developed as a "seamless bridge" between traditional FEM and CAD, while the appearance of NURBS in the curved VEM context is quite recent. On all this matter and without any pretence of exhaustivity we may quote, for example, [3, 4, 5, 18, 19, 28] with the references therein.

In the present paper we extend our approach for spline curvilinear polygons developed in [23], providing an algorithm (implemented in MATLAB) that computes a *low cardinality PI-type* (Positive weights and Interior nodes) cubature rule of Algebraic Degree of Exactness $ADE = n$, with at most $(n+1)(n+2)/2 = \dim(\mathbb{P}_n^2)$ nodes, over curvilinear polygons defined by piecewise rational functions (here and below $\dim(\mathbb{P}_n^d)$ denotes the space of d-variate polynomials with total degree not exceeding n). Typical examples are domains whose boundary is defined piecewise by NURBS curves or by composite Bezier curves. The key tools are a somehow overlooked theorem by Wilhelmsen [27] on *Tchakaloff sets* (Section 3), that are constructed by a specific *in-domain* algorithm for such curvilinear polygons developed in [24] (recalled in Section 2), and the sparse nonnegative solution of underdetermined moment matching systems by the Lawson-Hanson NonNegative Least Squares solver (Sections 3 and 4). Several numerical examples showing the effectiveness of the method are discussed in Section 5, where we also briefly describe the core of the freely available codes at [21].

We guess that this new approach could be useful in the NEFEM and curved VEM frameworks, quoted above as a meaningful motivation. In this respect, we stress that differently from the cubature method [19] developed in the NEFEM framework for NURBS curved elements, that is based on partition into curved triangles and a classical mapping approach, our approach is completely *triangulation-free* and the rules are *exact* up to polynomial degree n (to machine-precision) and of possibly lower cardinality. Moreover, *interiority* of the cubature nodes is guaranteed, differently from the recent paper [13] that treats cubature on NURBS-shaped domains by a direct use of Green's formula (here used only to compute the polynomial basis moments), generating however exterior nodes in some instances. Indeed, while the present paper is devoted to provide a general-purpose cubature method on NURBS-shaped domains, potentially useful in a variety of application including for example industrial CAGD as quoted in [13], we plan to develop the application to numerical modelling by FEM/VEM in future work.

2 An in-domain routine for rational spline curvilinear polygons

A key tool of our approach is an efficient implementation of the indicator function of planar NURBS-shaped domains, recently developed in [24]. For the reader's convenience, we recall in this section the main features of the corresponding algorithm. Indeed, we describe below an *in-domain* routine, for Jordan domains $\mathcal{S} \subset \mathbb{R}^2$,

1. whose boundary $\partial\mathcal{S}$ is described by parametric equations $x = \tilde{x}(t)$, $y = \tilde{y}(t)$, $t \in [a,b]$, $\tilde{x}, \tilde{y} \in C([a,b])$, $\tilde{x}(a) = \tilde{x}(b)$ and $\tilde{y}(a) = \tilde{y}(b)$;
2. for which there are partitions $\{I^{(k)}\}_{k=1,\dots,M}$ of $[a,b]$, and $\{I_j^{(k)}\}_{j=1,\dots,m_k}$ of each $I^{(k)} \equiv [t^{(k)}, t^{(k+1)}]$, such that the restrictions of $\tilde{x}$, $\tilde{y}$ on each closed interval $I^{(k)}$ are *rational splines*, w.r.t. the subintervals $\{I_j^{(k)}\}_{j=1,\dots,m_k}$.

   As notation, we will adopt

$$\tilde{x}(t) = \frac{u_{k,1}(t)}{v_{k,1}(t)},\ \tilde{y}(t) = \frac{u_{k,2}(t)}{v_{k,2}(t)},\ t \in I^{(k)}$$

being $u_{k,1}$, $u_{k,2}$, $v_{k,1}$, $v_{k,2}$ splines on $I^{(k)}$, sharing the same knots and having degree, respectively, $\eta_{k,1}$, $\eta_{k,2}$, $\delta_{k,1}$, $\delta_{k,2}$.

Observe that since $\tilde{x}$, $\tilde{y}$ belong to $C([a,b])$, the denominators $v_{k,1}$, $v_{k,2}$, $k = 1,\dots,M$ will be everywhere not null in the closed interval $I^{(k)}$.

In [23], the typical instances were spline curvilinear regions. Given the *vertices* $V_k \in \mathbb{R}^2$, $k = 1,\dots,M$ of the integration domain $\mathcal{S}$, then $\partial\mathcal{S} := \cup_{k=1}^{M} V_k \frown V_{k+1}$ (with the convention that $V_{M+1} = V_1$) and each curvilinear side $V_k \frown V_{k+1}$ was tracked by a parametric spline of degree δ_k, interpolating an ordered subsequence of knots $P_{1,k} = V_k, P_{2,k}, \dots, P_{m_k-1,k}, P_{m_k,k} = V_{k+1}$ with a suitable parametrization determining each $I_j^{(k)}$ (and thus each $I^{(k)}$). This structure of the boundary included also the case in which it could be defined by *composite Bezier closed curves*, where the k-th curve is of the form

$$\mathcal{B}(\tilde{t}) = \mathcal{B}(\omega_k(t)) = \sum_{i=0}^{m_k-1} b_{i,m_k-1}(t) P_{i+1,k},$$

where $\tilde{t} = \frac{t^{(k+1)}+t^{(k)}}{2} + \frac{t^{(k+1)}-t^{(k)}}{2} t := \omega_k(t), t \in [0,1]$ and

$$b_{i,l}(t) = \binom{l}{i} t^i (1-t)^{l-i},\ i = 0,\dots,l-1,\ \ t \in [0,1]$$

are known as *Bernstein polynomials*.

In this paper we generalise the previous framework, encompassing domains in which the boundary is locally a p-th degree NURBS curve [16, p.117], i.e. defined in the curvilinear side $V_k \frown V_{k+1}$ as

$$C(t) = \frac{\sum_{i=1}^{m_k} B_{i,p}(t)\,\lambda_{i,k} P_{i,k}}{\sum_{i=1}^{m_k} B_{i,p}(t)\,\lambda_{i,k}},\ t \in [t^{(k)}, t^{(k+1)}]$$

where $\{P_{i,k}\}_{i=1}^{m_k}$ are the *control points*, $\{\lambda_{i,k}\}_{i=1}^{m_k}$ are the weights and $\{B_{i,p}\}_{i=1}^{m_k}$ are the p-th degree B-spline basis functions [7, p.87] defined on the nonperiodic (and nonuniform) knot vector

$$U = \{\underbrace{t^{(k)},\dots,t^{(k)}}_{p+1}, t_{p+1}^{(k)}, \dots, t_{m_k-(p+1)}^{(k)}, \underbrace{t^{(k+1)},\dots,t^{(k+1)}}_{p+1}\}.$$

with $t^{(k)}_{p+j} \leq t^{(k)}_{p+j+1}$, $j = 1,\dots,m_k - 1$.

In order to define an algorithm that establishes if a point $P \in \mathbb{R}^2$ is inside (or not inside) such a $\mathcal{S}$, having in mind a variant of the procedure proposed in [23], for the reader's sake we describe it again, pointing out the needed modifications to these new instances.

A classical in-domain strategy is based on the well-known *Jordan curve theorem* that states that a point P belongs to a Jordan domain $\mathcal{S}$ if and only if, having taken a point $P^* \notin \mathcal{S}$ then the segment $\overline{P^*P}$ crosses $\partial\mathcal{S}$ an odd number $c(P)$ of times. There may be many pathological cases, for instance when $\overline{P^*P}$ touches a vertex or when includes a segment of $\partial\mathcal{S}$.

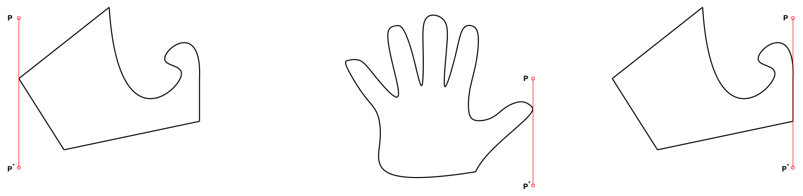

Fig. 2.1 Critical situations for establishing the crossing number on curvilinear polygons.

It can even happen that the boundary $\partial\mathcal{S}$ has a *critical* point $S = (\tilde{x}(\gamma), \tilde{y}(\gamma))$ where

$$\lim_{t\to\gamma^-} \tilde{x}'(t) \lim_{t\to\gamma^+} \tilde{x}'(t) < 0,$$

that geometrically means that there is locally a vertical turn of boundary from left to right (or conversely from right to left). In these cases, if the in-domain analysis of the point P is performed by means of *vertical* segments $\overline{P^*P}$ then the aforementioned Jordan theorem cannot be directly applied. These situations, illustrated in Figure 2.1, are not difficult to be ascertained, and we will initially require that $\overline{P^*P}$ does not contain any *critical point* or vertical side.

Under these assumptions, let $\mathcal{R}$ be a *cartesian* rectangle, i.e. with sides parallel to the cartesian axes, that *stricly* contains $\mathcal{S}$. Let P^* be the point of $\partial\mathcal{R}$ that shares the same abscissa P_x of $P = (P_x, P_y)$ but has ordinate strictly inferior of P_y. We intend to compute the *crossing number* $c(P)$, i.e. the number of times in which the vertical segment $\overline{P^*P}$ crosses $\partial\mathcal{S}$.

The first step consists in covering $\partial\mathcal{S}$ with the union of suitable rectangles $\mathcal{R}_1,\dots,\mathcal{R}_L$. Each $\mathcal{R}_j$, $j = 1,\dots,L$ contains a portion of $\partial\mathcal{S}$ that has no turning points and is parametrized by two *rational functions*, i.e. locally $(\tilde{x}(t),\ \tilde{y}(t))$ are the ratio of two polynomials. Once these $\mathcal{R}_j$, $j = 1,\dots,L$ are at hand, we will show that the evaluation of the crossing number $c(P)$ is straightfoward, requiring at most the solution of some polynomial equations.

In order to determine numerically these $\mathcal{R}_j$, we observe that since $\tilde{x}$, $\tilde{y}$ are rational splines in each $I^{(k)}$ then there are $I^{(k)}_j = [t^{(k)}_j, t^{(k)}_{j+1}] \subseteq I^{(k)}$, $k = 1,\dots,M$, $j =$

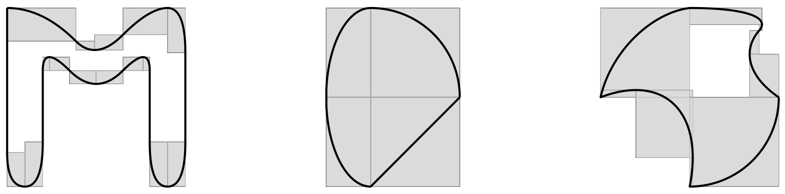

Fig. 2.2 Three domains $\mathcal{S}_3, \mathcal{S}_4, \mathcal{S}_5$, of curvilinear type, whose boundary may contain arc of circles, ellipses, segments as well as other NURBS blocks and its monotone boxes. Notice that monotone boxes may degenerate to a segment (as in the figure on the left, due to vertical segments) and distinct ones may overlap (as in the figure on the right).

$1,\dots,m_k-1$, where the restriction of $\tilde{x}$, $\tilde{y}$ to $I_j^{(k)}$ are rational functions, i.e.

$$\tilde{x}(t)=\frac{u_{k,j,1}(t)}{v_{k,j,1}(t)},\ \tilde{y}(t)=\frac{u_{k,j,2}(t)}{v_{k,j,2}(t)},\ t\in I_j^{(k)}$$

where $u_{k,j,1}$, $u_{k,j,2}$, $v_{k,j,1}$, $v_{k,j,2}$ are polynomials on $I_j^{(k)}$, having degree, respectively, $\eta_{k,1}$, $\eta_{k,2}$, $\delta_{k,1}$, $\delta_{k,2}$ (notice that they do not depend on j but just on the local degree of the splines $u_{k,1}$, $u_{k,2}$, $v_{k,1}$, $v_{k,2}$ in $I^{(k)}$).

If $\tilde{x}'$ changes sign in $(t_j^{(k)},t_{j+1}^{(k)})$, then define as $\mathcal{N}_j^{(k)}=\{t_i^{(j,k)}\}_{i=1,\dots,l_{j,k}}$ the set of all the points $t_i^{(j,k)}\in(t_j^{(k)},t_{j+1}^{(k)})$ such that $\tilde{x}'(t_i^{(j,k)})=0$ (notice that the restriction of $\tilde{x}$ to $I_j^{(k)}$ is a rational function with the denominator nowhere null, hence $\tilde{x}'$ exists), otherwise put $\mathcal{N}_j^{(k)}=\emptyset$. Next, let $T^{(j,k)}=\{t_j^{(k)},t_{j+1}^{(k)}\}\cup\mathcal{N}_j^{(k)}$, where we suppose that its elements, say $T_i^{(j,k)}$, are in increasing order. Observe that being $\tilde{x}(t)=u_{k,j,1}(t)/v_{k,j,1}(t)$, $t\in I_j^{(k)}$, the determination of the set $\mathcal{N}_j^{(k)}$ requires the solution of a polynomial equation. More precisely, since

$$\tilde{x}'(t)=\frac{u'_{k,j,1}(t)v_{k,j,1}(t)-u_{k,j,1}(t)v'_{k,j,1}(t)}{v^2_{k,j,1}(t)},\ t\in I_j^{(k)},$$

and $v^2_{k,j,1}(t)\neq 0$ for each $t\in I_j^{(k)}$, we get that $\tilde{x}'(t)=0$ if and only if

$$u'_{k,j,1}(t)v_{k,j,1}(t)-u_{k,j,1}(t)v'_{k,j,1}(t)=0,$$

and consequently the determination of $\mathcal{N}_j^{(k)}$ requires the solution of a polynomial equation of degree $\eta_{k,1}+\delta_{k,1}-1$.

Now define the rectangles $\mathcal{B}_i^{(j,k)}$, named *monotone boxes* in [23],

$$\mathcal{B}_i^{(j,k)}:=[\min_{t\in I_i^{(j,k)}}\tilde{x}(t),\ \max_{t\in I_i^{(j,k)}}\tilde{x}(t)]\times[\min_{t\in I_i^{(j,k)}}\tilde{y}(t),\ \max_{t\in I_i^{(j,k)}}\tilde{y}(t)].$$

where $I_i^{(j,k)} := [T_i^{(j,k)}, T_{i+1}^{(j,k)}]$. Observe that by definition if $\mathcal{N}_j^{(k)} = \emptyset$, then there is only the monotone box $\mathcal{B}_1^{(j,k)}$. Since $\tilde{y}$ restricted to $[T_i^{(j,k)}, T_{i+1}^{(j,k)}]$ is a rational function, the evaluation of

$$\min_{t \in [T_i^{(j,k)}, T_{i+1}^{(j,k)}]} \tilde{y}(t), \quad \max_{t \in [T_i^{(j,k)}, T_{i+1}^{(j,k)}]} \tilde{y}(t)$$

can be easily determined once the derivative of the polynomial $\tilde{y}'$ is at hand, by computing its zeros in $[T_i^{(j,k)}, T_{i+1}^{(j,k)}]$ and the evaluation of $\tilde{y}$ at $T_i^{(j,k)}$ and $T_{i+1}^{(j,k)}$.

At the end of this procedure we have determined $I_i^{(j,k)}$, so that

- the restrictions of $\tilde{x}$, $\tilde{y}$ to each $I_i^{(j,k)} \subseteq [a,b]$ are rational functions,
- $\tilde{x}$ is a monotone function (and consequently there are no turning points of $\partial\mathcal{S}$ in the interior of each box $\mathcal{B}_i^{(j,k)}$).

Once the set $\mathcal{B} := \{\mathcal{B}_i^{(j,k)}\}$ is available, we apply the crossing theorem to test whether or not $P = (P_x, P_y)$ is inside the domain $\mathcal{S}$. First we determine

$$\mathcal{B}(P) = \{B = [\alpha_1, \beta_1] \times [\alpha_2, \beta_2] \in \mathcal{B} : P_x \in [\alpha_1, \beta_1], P_y \geq \alpha_2\}.$$

Intuitively, $\mathcal{B}(P)$ takes into account all the monotone boxes $\mathcal{B}_l$ such that $\overline{P^*P} \cap \mathcal{B}_l \neq \emptyset$, and that may contribute to the evaluation of $c(P)$.

Consider a monotone box $\mathcal{B}_l = [\alpha_1^{(l)}, \beta_1^{(l)}] \times [\alpha_2^{(l)}, \beta_2^{(l)}] \in \mathcal{B}(P)$. If $P_y > \beta_2^{(l)}$ then the point P *does not belong* to the monotone box $\mathcal{B}_l$ and necessarily the segment $\overline{P^*P}$ crosses the boundary $\partial\mathcal{S}$ once in $\mathcal{B}_l$ and *below P*, due to the monotonicity of $\tilde{x}$ in $\mathcal{B}_l$. Otherwise, we have that $P \in \mathcal{B}_l$. Since by assumption $\overline{P^*P}$ does not contain a critical point or a vertical segment of the boundary and $\mathcal{B}_l$ includes a certain portion of $\partial\mathcal{S}$ described parametrically by two rational functions, say $\tilde{x}|_{\mathcal{B}_l}$, $\tilde{y}|_{\mathcal{B}_l}$, with arguments in the interval $I|_{\mathcal{B}_l} \subseteq [a,b]$, in which $\tilde{x}|_{\mathcal{B}_l}$ is monotone and such that $P_x \in \tilde{x}|_{\mathcal{B}_l}(I|_{\mathcal{B}_l})$, necessarily there is a unique root $t^* \in I|_{\mathcal{B}_l}$ of the polynomial equation $\tilde{x}|_{\mathcal{B}_l}(t) = P_x$. In particular, being $\tilde{x}|_{\mathcal{B}_l}(t) = \frac{u(t)}{v(t)}$, for certain polynomials u, v, then t^* is the unique solution in $I|_{\mathcal{B}_l}$ of the polynomial equation $u(t) - P_x \cdot v(t) = 0$.

Next,

- if $\tilde{y}(t^*) < P_y$ then the segment $\overline{P^*P}$ crosses the boundary $\partial\mathcal{S}$ once in the monotone box, *below P*;
- if $\tilde{y}(t^*) > P_y$ then the segment $\overline{P^*P}$ does not cross the boundary $\partial\mathcal{S}$ once in B, *below P*;
- if $\tilde{y}(t^*) = P_y$ then P is on the boundary $\partial\mathcal{S}$.

As result, in the assumptions mentioned above, counting all these crossings, we determine whether a point P is inside or not inside the domain $\mathcal{S}$.

When the vertical segment $\overline{P^*P}$ contains a *critical point* or a portion of a vertical side of $\partial\mathcal{S}$, we used an algorithm that is based on the well-known *winding* theorem to ascertain whether P belongs or not to $\mathcal{S}$. To this purpose, we computed by a (shifted) Gauss-Legendre rule of sufficiently high degree of exactness the so called *winding number*, $\text{wind}(P, \tilde{x}, \tilde{y}) \in \mathbb{Z}$,

$$\text{wind}(P, \tilde{x}, \tilde{y}) := \frac{1}{b-a} \int_a^b \frac{\tilde{y}'(t)(\tilde{x}(t) - P_x) - \tilde{x}'(t)(\tilde{y}(t) - P_y)}{(\tilde{x}(t) - P_x)^2 + (\tilde{y}(t) - P_y)^2} dt.$$

If the quantity $\mathrm{wind}(P,\tilde{x},\tilde{y})$ is odd then the point belongs to $\mathcal{S}$ otherwise is not inside such domain. We observe that the evaluation of $\mathrm{wind}(P,\tilde{x},\tilde{y})$ can be difficult when P is close to the boundary, but in general there is no need to compute such a quantity with high precision, in view of the fact that $\mathrm{wind}(P,\tilde{x},\tilde{y})$ is an integer.

Remark 2.1 In our implementation, each monotone box $\mathcal{B}_i^{(j,k)}$ has abscissae ranging in the interval

$$\left[\min(\tilde{x}(T_i^{(j,k)}),\tilde{x}(T_{i+1}^{(j,k)})),\max(\tilde{x}(T_i^{(j,k)}),\tilde{x}(T_{i+1}^{(j,k)}))\right].$$

Partitioning $[T_i^{(j,k)},T_{i+1}^{(j,k)}]$ as $\cup_{s=1}^{n_{i,j,k}-1}[T_{i,s}^{(j,k)},T_{i,s+1}^{(j,k)}]$ where

$$T_i^{(j,k)}=T_{i,1}^{(j,k)}<T_{i,2}^{(j,k)}<\ldots T_{i,n_{i,j,k}}^{(j,k)}=T_{i+1}^{(j,k)}$$

and defining the monotone boxes

$$\mathcal{B}_{i,s}^{(j,k)}:=[\min_{t\in I_{i,s}^{(j,k)}}\tilde{x}(t),\ \max_{t\in I_{i,s}^{(j,k)}}\tilde{x}(t)]\times[\min_{t\in I_{i,s}^{(j,k)}}\tilde{y}(t),\ \max_{t\in I_{i,s}^{(j,k)}}\tilde{y}(t)].$$

where $I_{i,s}^{(j,k)}=[T_{i,s}^{(j,k)},T_{i,s+1}^{(j,k)}]$, we have that $\partial\mathcal{S}\subset\cup_{i,s,j,k}\mathcal{B}_{i,s}^{(j,k)}\subseteq\mathcal{B}$.

In such a case, an inferior or equal number of polynomial equations is needed to establish whether or not a point P belongs to $\mathcal{S}$. In spite of this, we observe that there is numerical evidence that a too high number of monotone boxes does not improve the performance of the *in-domain* process.

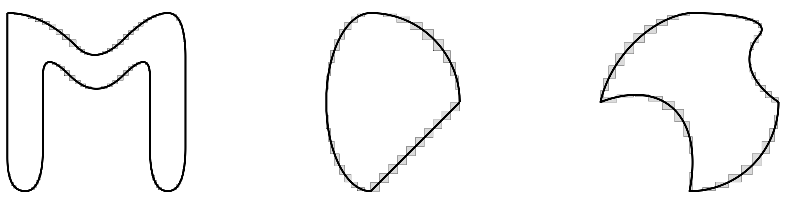

Fig. 2.3 Three domains $\mathcal{S}_3,\mathcal{S}_4,\mathcal{S}_5$, of curvilinear type, whose boundary may contain arc of circles, ellipses, segments as well as other NURBS blocks and is covered by a *larger* number of *monotone boxes* (compare with Figure 2.2).

Remark 2.2 In the implementation of the algorithm, to see if a point $P=(P_x,P_y)$ belongs to the boundary $\partial\mathcal{S}$, we solved a certain *polynomial* equation $u(t^*)-P_x\cdot v(t^*)=0$ and then tested that that its unique solution t^* satisfies $\tilde{y}(t^*)=P_y$. In view of numerical errors, we can only establish that a point is very close to $\partial\mathcal{S}$, that is $|\tilde{y}(t^*)-P_y|$ is below a certain tolerance fixed by the user.

For similar numerical issues, when a point $P=(P_x,P_y)$ has abscissa P_x close to that of a turning point or vertical side, we prefer to use the winding-algorithm.

3 On the construction of Tchakaloff-like cubature rules

Now, we show how to extract from a sufficiently dense discretization of a compact set a PI-type cubature rule with low cardinality (not exceeding the dimension of the polynomial space $\mathbb{P}_n^2$, being n the Algebraic Degree of Exactness). A fundamental existence result is the following version of Tchakaloff theorem [26, 17]:

Theorem 3.1 *Let μ be a positive measure on the compact domain $D \subset \mathbb{R}^d$ and let n be a positive integer. Then there are $\nu \leq dim(\mathbb{P}_n^d(D))$ points $\{Q_j\} \in D$ and positive real numbers $\{w_j\}$ such that*

$$\int_D p(P)\, d\mu = \sum_{j=1}^{\nu} w_j p(Q_j) \tag{3.1}$$

for all $p \in \mathbb{P}_n^d(D)$ (the space of d-variate polynomials of degree not exceeding n, restricted to D).

We aim to compute a set of nodes $\{Q_j\} \subset D$ and corresponding positive weights, satisfying (3.1). A key result is the following theorem by Wilhelmsen in [27], extending a proposition by Davis [6]:

Theorem 3.2 *Let $\mathcal{F}$ be the linear span of continuous, real-valued, linearly independent functions $\{f_k\}_{k=1,\dots,N}$ defined on a compact set $D \subset \mathbb{R}^d$. Assume that $\mathcal{F}$ satisfies the Krein condition (i.e. there is at least one $f \in \mathcal{F}$ which does not vanish on D) and that L is a positive linear functional on $\mathcal{F}$, i.e. $Lf > 0$ for every $f \in \mathcal{F}$, $f \geq 0$ not vanishing everywhere in D. If $\{P_i\}_{i=1}^{+\infty}$ is an everywhere dense subset of D, then for sufficiently large $\mathcal{I}$, the set $X = \{P_i\}_{i=1}^{\mathcal{I}}$ is a Tchakaloff set, i.e.*

$$Lf = \sum_{j=1}^{\nu} w_j f(Q_j),\ \forall f \in \mathcal{F} \tag{3.2}$$

where $w_j > 0\ \forall j$ and $\{Q_j\}_{j=1}^{\nu} \subset X \subset D$, with $\nu = card(\{Q_j\}) \leq N$.

Notice that we can apply this theorem to the case of algebraic cubature, setting $Lf = \int_D f(P)\ d\mu$ (being μ a positive measure on the compact set $D \subset \mathbb{R}^d$) and $\mathcal{F} = \mathbb{P}_n^d(D)$. Such a kind of formula can be named "*Tchakaloff-like algebraic cubature rule*".

Applying Theorem 3.2 to our instance, i.e. $D = \mathcal{S}$, and supposing that a Tchakaloff set X is at hand, given any polynomial basis $\{\phi_j\}$ of $\mathbb{P}_n^d$ define the Vandermonde-like matrix

$$V = V_n(X) = (\phi_j(P_i))_{i,j} \in \mathbb{R}^{\mathcal{I} \times N}\ . \tag{3.3}$$

Denoting by $\gamma = \{\gamma_j\}$ the vector of moments

$$\gamma_j = \int_{\mathcal{S}} \phi_j(P)\, d\mu\ ,\ \ j = 1, \dots, N\ , \tag{3.4}$$

the (underdetermined) $N \times \mathcal{J}$ moment system

$$V^T u = \gamma\,. \tag{3.5}$$

has a sparse nonnegative solution u to (3.5), whose nonvanishing components (i.e., the weights $\{w_j\} = \{u_i > 0\}$) are at most $N = \dim(\mathbb{P}_n^d)$. Furthermore, the nodes $\{Q_j\} \subset X$ are determined by the indices of the non-zero components of u (see e.g. [22]).

In the present work, we determine the vector u, solution of (3.5), by means of Lawson-Hanson iterative algorithm, that automatically seeks a sparse solution of the NonNegative Least Squares problem

$$\min_{u \geq 0} \|V^T u - \gamma\|_2 \tag{3.6}$$

by active set optimization; cf. [15]. We point out that there are several alternative implementations available in MATLAB, as the built-in function `lsqnonneg` or the open-source routine of the package `NNLSlab` in [20]. On the other hand, in the recent papers [9, 11] an acceleration of the Lawson-Hanson algorithm has been discussed in the framework of compression of discrete probability measures and regression, based on the concept of “Deviation Maximization” instead of standard column pivoting for the underlying QR factorizations [10]. Such a method, called LHDM, shows remarkable speed-ups and in perspective could be applied also in the cubature framework.

4 Implementing Tchakaloff-like algebraic cubature rules

In this section we describe a procedure that determines a Tchakaloff-like algebraic cubature formula with nodes in the integration domain $\mathcal{S}$, having $ADE = n$ over $\mathcal{S}$, i.e. such that

$$\int_{\mathcal{S}} p(x,y)\,dx\,dy = \sum_{j=1}^{\nu} w_j p(Q_j)\,,\ \ \nu \leq N = \dim(\mathbb{P}_n^2) = (n+1)(n+2)/2$$

for any bivariate polynomial $p \in \mathbb{P}_n^2$.

We assume that $I_s^{(k)} = [t_s^{(k)}, t_{s+1}^{(k)}]$, $k = 1,\dots,M$, $s = 1,\dots,m_k - 1$, is a partition of $[a,b]$ and that $\partial\mathcal{S} = \{(\tilde{x}(t),\tilde{y}(t)),\ t \in [a,b]\}$, where the restrictions of $\tilde{x}$, $\tilde{y} \in C([a,b])$ to each $I_s^{(k)}$ are rational functions.

The first step consists in computing the *moments* γ_j of a suitable polynomial basis $\{\phi_j\}$ of total degree n over $\mathcal{S}$, i.e.

$$\gamma_j = \int_{\mathcal{S}} \phi_j(x,y)\,dx\,dy\,,\ j = 1,\dots,N.$$

As in [23], in our MATLAB implementation we have adopted as polynomial basis $\{\phi_j\}$, $1 \leq j \leq N$, the lexicographically ordered total-degree product Chebyshev basis

$$\phi_j(x,y) = T_{h_1}(\alpha_1(x)) \cdot T_{h_2}(\alpha_2(y)),\ 0 \leq h_1 + h_2 \leq n$$

(here j is the position of (h_1,h_2) in the ordering) and $(x,y) \in \mathcal{R}^* = [a_1,b_1] \times [a_2,b_2]$ where

- $T_h(\cdot) = \cos(h \arccos(\cdot))$ is the h-degree Chebyshev polynomial of first kind;
- if $\mathcal{R}^* = [a_1, b_1] \times [a_2, b_2]$ is the *bounding box* of $\mathcal{S}$ (that corresponds to the smallest cartesian rectangle, i.e. with sides parallel to the axes, containing the domain $\mathcal{S}$, easily available when the all the monotone boxes are determined), then $\alpha_i(s) = (2s - b_i - a_i)/(b_i - a_i)$, $s \in [a_i, b_i]$, $i = 1, 2$; we point out that if the boundary $\partial\mathcal{S}$ is described by a NURBS curve, then $\mathcal{R}^*$ is the smallest cartesian rectangle containing the convex hull of the control points of the NURBS.

The choice of this basis comes from the necessity of avoiding the extreme ill-conditioning of Vandermonde matrices in the standard monomial basis. In virtue of Gauss-Green theorem (see e.g. [2]),

$$\gamma_j = \int_{\mathcal{S}} \phi_j(x,y)\, dx dy = \oint_{\partial\mathcal{S}} \Psi_j(x,y)\, dy \tag{4.1}$$

where

$$\Psi_j(x,y) = \int \phi_j(x,y)\, dx = T_{h_2}(\alpha_2(y)) \int T_{h_1}(\alpha_1(x))\, dx$$

To this purpose we observe that

$$\begin{aligned}
&\int T_0(\alpha_1(x))\, dx = x, \\
&\int T_1(\alpha_1(x))\, dx = \frac{b_1 - a_1}{4} \cdot \alpha_1^2(x), \\
&\int T_h(\alpha_1(x))\, dx = \frac{b_1 - a_1}{2} \cdot \left(\frac{h}{h^2 - 1} T_{h+1}(\alpha_1(x)) - \frac{x}{h-1} T_h(\alpha_1(x)) \right), \; h \geq 2.
\end{aligned}$$

If $P_{k,s} := (\tilde{x}(t_s^{(k)}), \tilde{y}(t_s^{(k)}))$ and $P_{k,s} \frown P_{k,s+1}$ is the arc of $\partial\mathcal{S}$ joining $P_{k,s}$ with $P_{k,s+1}$, one gets

$$\begin{aligned}
\gamma_j &= \oint_{\partial\mathcal{S}} \Psi_j(x,y)\, dy = \sum_{k,s} \int_{P_{k,s} \frown P_{k,s+1}} \Psi_j(x,y)\, dy \\
&= \sum_{k,s} \int_{t_s^{(k)}}^{t_{s+1}^{(k)}} \Psi_j(\tilde{x}(t), \tilde{y}(t))\, \tilde{y}'(t) dt \,.
\end{aligned} \tag{4.2}$$

The evaluation of the integrals on the right-hand side of (4.2) require some attention. In [23] it was considered the case in which $\tilde{x}$ and $\tilde{y}$ are in $[t_s^{(k)}, t_{s+1}^{(k)}]$ both polynomials of degree δ_k, observing that since Ψ_j is a polynomial of total degree $n+1$, each integrand in the last sum of (4.2) is a polynomial of degree $(n+1)\delta_k + \delta_k - 1 = (n+2)\delta_k - 1$ and consequently can be exactly integrated by a (shifted) Gauss-Legendre formula with $\lceil \frac{(n+2)\delta_k}{2} \rceil$ points.

The setting in which $\tilde{x}$ and $\tilde{y}$ are rational functions is more delicate, and requires for example the usage of appropriate high order Gauss-Legendre rules [14], or adaptive routines as the MATLAB built-in routine `integral`, or the Extended Rational Fejèr Quadrature Rules proposed in [8]. See also [19, §4] for several numerical tests and comparisons between methods.

The second step consists in extracting the nodes and positive weights of a Tchakaloff-like algebraic cubature rule. To this purpose, let $\mathcal{P}_0 = \emptyset$. We introduce a sequence of tensorial grids $\mathcal{M}_\ell$ in the rectangle $\mathcal{R}^* := [a_1, b_1] \times [a_2, b_2]$ containing $\mathcal{S}$, with $\mathcal{M}_\ell$ finer as ℓ increases, determining by the *in-domain* algorithm, at the ℓ-th iteration of the procedure, the set $\mathcal{P}_\ell = \mathcal{P}_{\ell-1} \cup (\mathcal{M}_\ell \cap \mathcal{S})$, that is the points of the previous meshes, as well as of the present one, belonging to the integration domain.

Next, we apply the Lawson-Hanson algorithm to attempt the extraction of the Tchakaloff formula with nodes $\{(x_i^{(\ell)}, y_i^{(\ell)})\}_{i=1,\dots,\nu_\ell}$ and corresponding positive weights $\{w_i^{(\ell)}\}_{i=1,\dots,\nu_\ell}$, $\nu_\ell \leq N$, finally testing if the so obtained rule is such that

$$\gamma_j^{(\ell)} = \sum_{i=1}^{\nu_\ell} w_i^{(\ell)} \phi_j(x_i^{(\ell)}, y_i^{(\ell)}),\ j = 1, \dots, N\ ,$$

well approximates the set of moments $\gamma = \{\gamma_j\}$, i.e.

$$\|\gamma^{(\ell)} - \gamma\|_2 \leq \varepsilon \tag{4.3}$$

where ε is a tolerance fixed by the user. If (4.3) does not hold we iterate the procedure until (4.3) is satisfied or a maximum number of iterations is reached, providing in this case an error message. It is important to observe that in exact arithmetic this procedure has finite termination in view of a theorem by Wilhelmsen mentioned above [27], since the set $\mathcal{P}_\ell$ becomes sufficiently dense after a finite number of iterations.

5 Numerical experiments

In this section, we test either the in-domain routine, either the cubature algorithm that produces a *Tchakaloff-like algebraic cubature rule* with nodes belonging to the integration domain $\mathcal{S}$ whose boundary $\partial\mathcal{S}$ is defined via piecewise rational functions.

In particular, we intend to determine these rules when $\partial\mathcal{S}$ is described by piecewise NURBS curves, implying that each curvilinear side $V_k \frown V_{k+1} \subseteq \partial\mathcal{S}$, $k = 1, \dots, M$, is defined parametrically as

$$C(t) = \frac{\sum_{i=1}^{m_k} B_{i,p}(t)\, \lambda_{i,k} P_{i,k}}{\sum_{i=1}^{m_k} B_{i,p}(t)\, \lambda_{i,k}},\ t \in I^{(k)} := [t^{(k)}, t^{(k+1)}]$$

where

- $P_{i,k} = ((P_{i,k})_x, (P_{i,k})_y)$, $i = 1, \dots, m_k$, $k = 1, \dots, M$, are the *control points*,
- $\{\lambda_{i,k}\}_{i=1}^{m_k}$, $k = 1, \dots, M$, are the weights,
- $\{B_{i,p}\}_{i=1}^{m_k}$ are the p-th degree B-spline basis functions [7, p.87] defined on the nonperiodic (and nonuniform) knot vector

$$U = \{\underbrace{t^{(k)}, \dots, t^{(k)}}_{p+1}, t_{p+1}^{(k)}, \dots, t_{m_k-(p+1)}^{(k)}, \underbrace{t^{(k+1)}, \dots, t^{(k+1)}}_{p+1}\}.$$

with $t_{p+j}^{(k)} \leq t_{p+j+1}^{(k)}$, $j = 1, \dots, m_k - 1$, $k = 1 \dots, M$.

In order to satisfy the assumptions introduced in Section 2, we first need to

1. determine the splines $u_{k,1}$, $u_{k,2}$, $v_{k,1}$, $v_{k,2}$ on $I^{(k)}$ defining

$$\tilde{x}(t) = \frac{u_{k,1}(t)}{v_{k,1}(t)},\ \tilde{y}(t) = \frac{u_{k,2}(t)}{v_{k,2}(t)},\ t \in I^{(k)},$$

i.e.

$$u_{k,1}(t) = \sum_{i=1}^{m_k} B_{i,p}(t)\,\lambda_{i,k}(P_{i,k})_x,\ u_{k,2}(t) = \sum_{i=1}^{m_k} B_{i,p}(t)\,\lambda_{i,k}(P_{i,k})_y,$$
$$v_{k,1}(t) = v_{k,2}(t) = \sum_{i=1}^{m_k} B_{i,p}(t)\,\lambda_{i,k}; \tag{5.1}$$

2. convert from a B-spline notation to a piecewise one via MATLAB command `fn2fm`.

Once we have described these rational splines in piecewise form, we can determine, as introduced in the previous sections, the in-domain and the cubature routine.

The latter is implemented by the MATLAB code `cubRS`, in which

1. we define a mesh on the smallest rectangle $\mathcal{R}^* = [a_1,b_1] \times [a_2,b_2]$ containing the domain $\mathcal{S}$; in particular for $ADE = n$, then setting $\tau = \lfloor n^{1.5} \rfloor$, we considered the points $P_{ij} = (x_i, y_j)$ where

$$x_i = a_1 + i\frac{a_2 - a_1}{\tau - 1},\ y_j = b_1 + j\frac{b_2 - b_1}{\tau - 1},\ \ 0 \le i,j \le \tau - 1$$

i.e. a uniform tensor grid mesh $\mathcal{M}_1$, based on τ equispaced points in each direction (such a choice of τ is based on numerical experiments, in order to try to keep low the number of grid refinements);
2. we determine the points of $\mathcal{M}_1$ strictly inside the spline curvilinear polygon $\mathcal{S}$, say $\mathcal{P}_1 \subseteq \mathcal{M}_1$, by the *in-domain* algorithm developed in Section 2 and implemented by the MATLAB code `inRS`;
3. we compute the moments over $\mathcal{S}$ of the n-th total-degree product Chebyshev basis

$$\{T_{h_1}(\alpha_1(x))T_{h_2}(\alpha_2(y))\},\ (x,y) \in [a_1,b_1] \times [a_2,b_2],\ 0 \le h_1 + h_2 \le n$$

with $\alpha_i(s) = (2s - b_i - a_i)/(b_i - a_i)$, $i = 1,2$, by means of Gauss-Green theorem and suitable quadrature rules along the rational spline boundary arcs;
4. we extract the nodes of a PI-type formula with $ADE = n$ from $\mathcal{P}_1$ and determine the relative weights using the Lawson-Hanson algorithm on the corresponding NNLS problem, as described at the end of Section 3; cf. (3.5)-(3.6).

In case the cubature rule is not satisfactory, i.e. the 2-norm of the moment error is larger than a fixed tolerance, say $\varepsilon = 10^{-12}$ (cf. (4.3)), we proceed iteratively by defining finer uniform tensor grids $\mathcal{M}_\ell$ (increasing the value of τ, as $\tau_{\ell+1} = \lfloor \beta\tau_\ell \rfloor$ with e.g. $\beta = 1.5$), determining at the ℓ-th iteration, with $\ell > 1$, those points belonging to $\mathcal{S}$, say $\mathcal{P}_\ell \subseteq \mathcal{M}_\ell$, and performing step 4 of the algorithm above with the set $\bigcup_{i=1}^{\ell} \mathcal{P}_i$ instead of $\mathcal{P}_1$.

Moreover, these algebraic Tchakaloff rules of PI-type

– are optimally stable (positive weights), i.e. the cubature condition number

$$\text{cond}(\{w_i\}) = \frac{\sum_{i=1}^{\nu} |w_i|}{|\sum_{i=1}^{\nu} w_i|}$$

is equal to 1, the best possible result for rules with $ADE \geq 0$;
– have cardinality particularly low (though usually not minimal);
– are suitable when sampling of the integrand is not possible outside the domain.

As in [23], the possible ill-conditioning of Chebyshev-Vandermonde matrices may harm the extraction procedure. Thus we *mitigated* these instances by a suitable discrete orthogonalization of the polynomial basis via the *economy size* QR factorization (as described for example in [22]). All the routines for the numerical experiments are available at [20] and have been tested on a computer with a M1 chip, with 16 GB of RAM, running MATLAB R2021b.

We start our numerical tests by considering a convex domain $\mathcal{S}_1$ and a non-convex domain $\mathcal{S}_2$, corresponding respectively to the intersection and the difference of a convex polygonal element with a disk, whose boundary $\partial\mathcal{S}_i$, $i = 1,2$, is composed by an arc of circle and a polygonal arc with 6 sides. Such curvilinear elements can arise for example within VEM application in computational mechanics, when a circular hole in a plate or a circular inclusion in a fibre-reinforced material are present (cf. [4]). Since an arc of a circle and a polygonal arc can be exactly described with piecewise NURBS curves, an algebraic cubature rule can be determined on $\mathcal{S}_1$ and $\mathcal{S}_2$ by means of the approach used in the present paper.

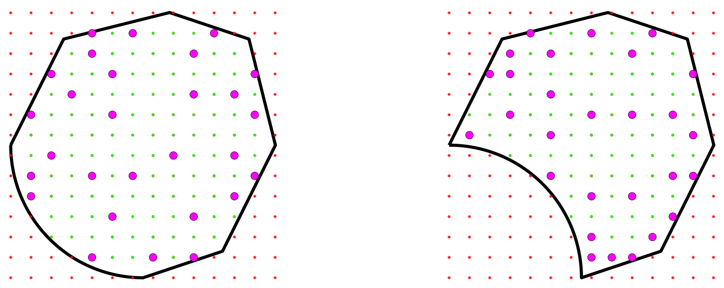
Fig. 5.1 The curvilinear domains $\mathcal{S}_i$ with $i = 1,2$, the grid points P outside the domain or on its boundary (in red), those inside the domain (in green) and the nodes of a cubature formula of PI-type for $n = 6$ (28 magenta dots).

Relatively to these domains, we compute the maximum relative error of the moments of the lexicographically ordered monomial basis of total degree n, $\{x^{j_1}y^{j_2}\}$, $0 \leq j_1 + j_2 \leq n$, with the new general purpose algorithm as well as with that used in [4], introduced for producing a Tchakaloff-type algebraic cubature rule in the case of curvilinear polygons where one side of the boundary is a convex or concave arc of

	n	#	trial pts	cond	moment r.d.	cpu	cpu [4]
$\mathcal{S}_1$	2	6	76 (121)	1	$8e{-}16$	$4.7e{-}3$s	$5.3e{-}4$s
	4	15	76 (121)	1	$4e{-}15$	$4.8e{-}3$s	$2.3e{-}3$s
	6	28	122 (196)	1	$4e{-}15$	$5.4e{-}3$s	$1.7e{-}3$s
	8	45	325 (484)	1	$6e{-}15$	$7.9e{-}3$s	$4.7e{-}3$s
	10	66	681 (961)	1	$8e{-}15$	$1.5e{-}2$s	$8.3e{-}3$s
$\mathcal{S}_2$	2	6	64 (121)	1	$5e{-}16$	$3.3e{-}3$s	$1.6e{-}3$s
	4	15	64 (121)	1	$3e{-}15$	$3.9e{-}3$s	$1.8e{-}3$s
	6	28	99 (196)	1	$3e{-}15$	$5.1e{-}3$s	$2.6e{-}3$s
	8	45	265 (484)	1	$3e{-}15$	$6.8e{-}3$s	$3.7e{-}3$s
	10	66	555 (961)	1	$6e{-}15$	$1.3e{-}2$s	$8.6e{-}3$s

Table 5.1 Degrees of exactness n, cardinality # of the extracted nodes, number of points used to extract the nodes versus all those generated in the bounding box (the latter in parentheses), cubature conditioning `cond`, maximum relative difference `moment r.d.` on the computation of the monomial basis on domain $\mathcal{S}_i$, $i = 1,2$, of the general purpose method adopted in this paper with that introduced in [4] and the respective median cputimes on 100 tests.

a circle. In particular, applying these procedures, for each n we run 100 tests finally taking the median of the cputimes.

The results summarized in Table 5.1 show that

– the new method produces numerically the same moments with a relative difference close to machine precision;
– though as expected the performance is a little worse, the cputimes are comparable, making these new rules attracting for cubature within FEM/VEM;
– both methods produce algebraic rules with positive weights.

The substantial advantage of the present approach is that it can be applied to much more general instances than [4], where the curved edges are circular arcs.

In order to show the flexibility of our method, we consider the domains depicted in Figure 2.2, that are from left to right,

1. a ”M” shaped domain $\mathcal{S}_3$, in which $\partial\mathcal{S}_3$ is determined by a unique order 3 NURBS curve with 16 distinct control points,
2. a convex domain $\mathcal{S}_4$, where $\partial\mathcal{S}_4$ is obtained by joining a circular and an elliptical arc, followed by a segment,
3. a concave domain $\mathcal{S}_5$ whose boundary $\partial\mathcal{S}_5$ consists of a unique NURBS curve of order 3 with 9 distinct control points.

We report in Table 5.2 the results that we obtained applying the new algorithm to the computation of algebraic cubature rules in which the degrees of exactness are $n = 2,4,6,8,10$.

In particular

– the column `#trial pts` addresses the number of points inside the domain used to extract the rule and all those tested in the bounding box (including some outside the domain and thus discarted); by construction all the points are internal to the domain;

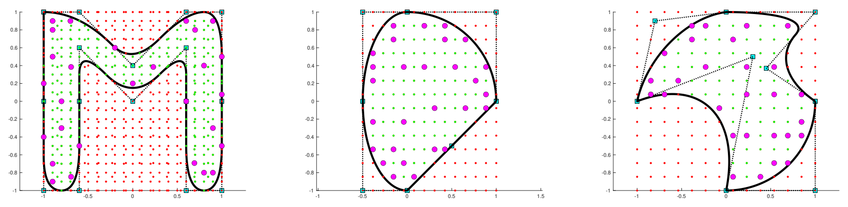

Fig. 5.2 The curvilinear domains $\mathcal{S}_i$ with $i = 3,4,5$, the grid points P outside the domain or on its boundary (in red), those inside the domain (in green) and the nodes of a cubature formula of PI-type for $n = 6$ (28 magenta dots). The control points of the NURBS curve are represented as cyan squares, joined to represent the so called *control points polygon*.

	n	#	# trial pts	cond	moment res	cpu
$\mathcal{S}_3$	2	6	28 (121)	1	$5e{-}16$	$1.3e{-}2$
	4	15	108 (377)	1	$1e{-}15$	$1.8e{-}2$
	6	28	225 (637)	1	$1e{-}15$	$2.2e{-}2$
	8	45	693 (1573)	1	$3e{-}15$	$3.4e{-}2$
	10	66	1304 (3077)	1	$5e{-}15$	$8.5e{-}2$
$\mathcal{S}_4$	2	6	65 (121)	1	$8e{-}16$	$4.8e{-}3$
	4	15	65 (121)	1	$2e{-}15$	$4.8e{-}3$
	6	28	109 (196)	1	$2e{-}15$	$6.6e{-}3$
	8	45	274 (484)	1	$2e{-}15$	$9.0e{-}3$
	10	66	609 (961)	1	$3e{-}15$	$1.5e{-}2$
$\mathcal{S}_5$	2	6	50 (121)	1	$5e{-}16$	$5.3e{-}3$
	4	15	50 (121)	1	$7e{-}16$	$6.1e{-}3$
	6	28	89 (196)	1	$1e{-}15$	$7.6e{-}3$
	8	45	239 (484)	1	$2e{-}15$	$1.1e{-}2$
	10	66	491 (961)	1	$4e{-}15$	$1.6e{-}2$

Table 5.2 Degree of precision n of the rule, cardinality # of the extracted nodes, cubature conditioning and moment residual of the rule on domains $\mathcal{S}_i$, $i = 3,4,5$, number of trial points used in the extraction, cubature condition number `cond`, moment residual of the rule and median of the cputime over 50 tests.

– the fact that in all tests the cubature conditioning is equal to 1, means that the rules have positive weights and thus optimal stability;
– the column `moment res` consists of the quantity $\|\gamma - \gamma^{(num)}\|_2$, where $\gamma = \{\gamma_j\}$ are the moments of the chosen shifted product Chebyshev basis of degree n, while $\gamma^{(num)} = \{\gamma_j^{(num)}\}$ consists in their evaluation using the cubature rule provided by the algorithm; their matching close to machine precision confirm that the rules have (numerical) algebraic degree of exactness equal to n;
– the column with the cputimes required by the algorithm to compute the rule, displays the median over 50 tests; we observe, that as seen before for domains $\mathcal{S}_1$, $\mathcal{S}_2$, they are still fast being at most of the order of some 10^{-2} seconds, depending on the complexity of the domain.

In Figure 5.3 we illustrate the performance of these new cubature rules to determine

$$I = \int_{\mathcal{S}_i} (c_0 + c_1 x + c_2 y)^n \, dx\, dy \;,\;\; i = 3,4,5 \;,$$

making 100 trials with uniform random coefficients $c_j \in (0,1)$, $j = 0,1,2$ and $ADE = n$, $n = 2,4,6,8,10$. The reference values of these integrals have been determined by applying Gauss-Green theorem and Gauss-Legendre high-order quadrature along the rational spline boundary $\partial\mathcal{S}$. We have plotted with a dot the relative error RE_k made by the rule (log scale) and by a larger circle the logarithmic average on all the trials, i.e. $\sum_{k=1}^{100} \log(RE_k)/100$.

The tests show that in spite of the fact that the moments are computed close to machine precision, there is a little deterioration of the logarithmic average error for $n = 10$, while for lower degrees this value remains is in general lower than 10^{-14}. Notice that an ADE greater than 10 is usually beyond what is needed for example in VEM applications.

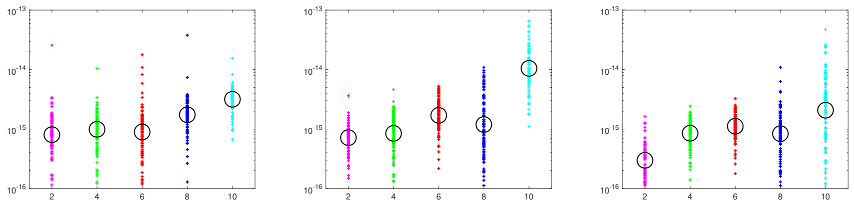

Fig. 5.3 *Dots*: Relative errors RE_k, $k = 1,\dots,100$, on cubature over random polynomials $(c_0 + c_1 x + c_2 y)^n$ on $\mathcal{S}_3$ (on the left), $\mathcal{S}_4$ (in the middle) and $\mathcal{S}_5$ (on the right). *Circles*: average logarithmic error, i.e. $10^{\sum_{k=1}^{100} \log(RE_k)/100}$. The abscissae are the ADE of the formula and are equal to $2,4,6,8,10$.

As a further illustration, we report in Table 5.3 the relative errors made by the Tchakaloff-like rules when approximating $\int_{\mathcal{S}_i} f_k(x,y)\, dx\, dy$, where

$$\begin{aligned} f_1(x,y) &= \exp(-(x^2+y^2)), \\ f_2(x,y) &= ((x-x_0)^2+(y-y_0)^2)^{11/2} \;,\; (x_0,y_0) = (0,0.4), \\ f_3(x,y) &= ((x-x_0)^2+(y-y_0)^2)^{1/2} \;,\; (x_0,y_0) = (0,0.4), \end{aligned}$$

that are examples of functions with different degree of regularity on each domain $\mathcal{S}_i$, $i = 3,4,5$. The reference values of these integrals are those obtained by the same routines with $ADE = 20$. As expected, in both the domains the quality of the approximation worsens for less regular integrands (indeed $f_1 \in C^\infty(\mathcal{S}_i)$, whereas $(0,0.4) \in \mathcal{S}_i$ is a singular point for the first derivatives of f_3 and for 6-th derivatives of f_2).

As additional information, we show in Table 5.4 the median cputimes over 10 tests necessary to process # in-domain operations, on the regions $\mathcal{S}_i$, $i = 3,4,5$. These random points belong to the bounding box of each domain.

	$\mathcal{S}_3$			$\mathcal{S}_4$			$\mathcal{S}_5$		
ADE	f_1	f_2	f_3	f_1	f_2	f_3	f_1	f_2	f_3
2	$2e{-}02$	$4e{-}01$	$4e{-}02$	$4e{-}03$	$9e{-}01$	$6e{-}02$	$6e{-}03$	$2e{-}01$	$1e{-}02$
4	$3e{-}03$	$2e{-}01$	$9e{-}02$	$3e{-}04$	$2e{-}02$	$4e{-}02$	$9e{-}04$	$2e{-}01$	$1e{-}02$
6	$3e{-}04$	$4e{-}02$	$6e{-}03$	$4e{-}05$	$3e{-}02$	$2e{-}02$	$4e{-}05$	$1e{-}02$	$4e{-}03$
8	$3e{-}05$	$3e{-}03$	$2e{-}03$	$1e{-}06$	$8e{-}04$	$1e{-}03$	$2e{-}06$	$2e{-}03$	$3e{-}03$
10	$1e{-}06$	$8e{-}05$	$1e{-}03$	$8e{-}09$	$4e{-}05$	$2e{-}04$	$8e{-}08$	$3e{-}05$	$2e{-}04$

Table 5.3 Relative errors of the new rules on the domains $\mathcal{S}_i$, $i = 3,4,5$ with $ADE = 2,4,6,8,10$.

#	$\mathcal{S}_3$	$\mathcal{S}_4$	$\mathcal{S}_5$
10^2	$2.4e{-}03s$	$1.2e{-}03s$	$1.1e{-}03s$
10^3	$4.5e{-}03s$	$1.9e{-}03s$	$1.7e{-}03s$
10^4	$2.7e{-}02s$	$7.9e{-}03s$	$8.3e{-}02s$
10^5	$2.8e{-}01s$	$6.0e{-}02s$	$6.6e{-}02s$
10^6	$2.2e{+}00s$	$7.5e{-}01s$	$8.0e{-}01s$

Table 5.4 Median cputime over 30 tests of the application of the *in-domain* algorithm to # random points, in domains $\mathcal{S}_i$, $i = 3,4,5$.

Remark 5.1 All the MATLAB routines and demos are collected in the toolbox `CUB_RS` and can be freely downloaded at the homepage [21]. We point out that at the time of writing this paper, we are not aware of the existence of an official built-in NURBS toolbox (though it can be retrieved by third-parties), though MATLAB has a specific environment for rational splines. Thus we have been forced to implement a set of routines, in which the boundary $\partial\mathcal{S}$ of each domain can be described by piecewise rational splines, as in the case of parametric splines or composite Bezier curves or NURBS.

To this purpose, we used *structured arrays*, in which the k-th component contains the relevant data of the k-th component of curve $V_k \frown V_{k+1}$, that are control points, weights, order, knots, type of the curve (e.g. NURBS, composite Bezier curve or spline). Next, we provide the routines

– `inRS` that implement the in-domain algorithm described above,
– `cubRS` that computes a PI-type Tchakaloff-like algebraic cubature rule of exactness degree n,

for the designed domain $\mathcal{S}$.

Hoping that this software could be useful to the community, we wrote many demos that show how to define the boundary of the integration domain $\mathcal{S}$, how to use the in-domain routine `inRS` and the cubature rule generator `cubRS`.

6 Conclusion

We have implemented the construction of low-cardinality Positive Interior cubature rules over curvilinear polygons whose boundary is given by rational parametric curves, in particular NURBS curves. The method, which generalizes what we previously implemented for spline boundaries [23], relies on Davis-Wilhelmsen theorem about Tchakaloff-like representation of positive functionals on polynomial spaces, by sparse sampling on sufficiently dense sequences [26, 6, 27].

Computation of nodes and weights is accomplished by NNLS solution to an underdetermined moment-matching system, where the moments are obtained by Gauss-Green theorem, integrating via Gaussian quadrature suitable antiderivatives of the bivariate product Chebyshev basis for total-degree polynomials.

One of the main difficulties is the construction of sufficiently dense sequences, e.g. Halton sequences, in the domain, where we used our very recent implementation of the indicator function of NURBS-shaped domains via a covering of the boundary by "monotone boxes" and an economy use of the crossing number [24]. On the other hand, the moment-matching NNLS problem is coped via the classical Lawson-Hanson active-set algorithm, which naturally seeks a sparse solution, or one of its recent variants [15, 9, 11, 20].

The resulting algorithm, that we may call TDW-cubature (Tchakaloff-Davis-Wilhelmsen), together with its open-source Matlab implementation, turns out to be rather efficient, and potentially useful in several applications, for example within the emerging fields of FEM/VEM with curved elements. Indeed, we have recently adopted substantially the same approach for tetrahedralization-free cubature on general polyhedral elements [25], and even for the compression of QMC cubature on complicated 2D and 3D domains [12].

Acknowledgements

Work partially supported by the DOR funds and the biennial project BIRD 192932 of the University of Padova, and by the INdAM-GNCS 2022 Project "Methods and software for multivariate integral models". This research has been accomplished within the RITA "Research ITalian network on Approximation", and within the UMI Group TAA "Approximation Theory and Applications" (A. Sommariva).

References

1. Aldakheel, F., Hudobivnik, B., Artioli, E., Beirão da Veiga, L., Wriggers, P.: Curvilinear Virtual Elements for Contact Mechanics. Comput. Methods Appl. Mech. Eng. **372** (2020)
2. Apostol, T.M.: Calculus, 2nd ed., vol. II, Blaisdell (1969)
3. Artioli, E., Beirão da Veiga,L., and Dassi, F.: Curvilinear Virtual Elements for 2D solid mechanics applications. Comput. Methods Appl. Mech. Eng. **359** (2020)
4. Artioli, E., Sommariva, A., Vianello, M.: Algebraic cubature on polygonal elements with a circular edge. Comput. Math. Appl. **79**, 2057–2066 (2020)
5. Beirão da Veiga, L., Russo, A., Vacca, G.: The Virtual Element Method with curved edges. ESAIM Mathematical Modelling and Numerical Analysis **53**, 375–404 (2019)
6. Davis, P.J.: A construction of nonnegative approximate quadratures. Math. Comp. **21**, 578–582 (1967)

7. de Boor, C.: A Practical Guide to Splines, Rev.ed. Springer-Verlag, New York (2001)
8. Deckers, K., Mougaida, A., Belhadjsalah, H.: Algorithm 973: Extended Rational Fejér Quadrature Rules based on Chebyshev Orthogonal Rational Functions. ACM Transactions on Mathematical Software, **43**(4), 1–29 (2017)
9. Dell'Orto, M., Dessole, M., Marcuzzi, F.: The Lawson-Hanson Algorithm with Deviation Maximization: Finite Convergence and Sparse Recovery. Numer. Linear Algebra Appl., accepted upon revision. arXiv:2108.05345v3 (2022)
10. Dessole, M., Marcuzzi, F.: Deviation maximization for rank-revealing QR factorizations. Numer. Algorithms **91**, 1047–1079 (2022)
11. Dessole, M., Marcuzzi, F., Vianello, M.: Accelerating the Lawson-Hanson NNLS solver for large-scale Tchakaloff regression designs. Dolomites Res. Notes Approx. DRNA **13**, 20–29 (2020)
12. Elefante, G., Sommariva, A., Vianello, M.: CQMC: an improved code for low-dimensional Compressed Quasi-MonteCarlo cubature. Dolomites Res. Notes Approx. DRNA **15**, 92–100 (2022) (Special Issue “Software for Approximation 2022”)
13. Gunderman, D., Weiss, K., Evans, J.A.: Spectral mesh-free quadrature for planar regions bounded by rational parametric curves. Computer-Aided Design **130** (2021)
14. Hale, H., Townsend, A.: Fast and Accurate Computation of Gauss-Legendre and Gauss-Jacobi Quadrature Nodes and Weights. SIAM J. Sci. Comput. **35**(2), A652–A674 (2013)
15. Lawson, C.L., Hanson, R.J.: Solving least squares problems, Classics in Applied Mathematics 15, SIAM, Philadelphia (1995)
16. Piegl L.: The NURBS Book, 2nd Edition, Springer-Verlag, Berlin-Heidelberg (1997)
17. Putinar, M.: A note on Tchakaloff theorem. Proc. Amer. Math. Soc. **125**, 2409–2414 (1997)
18. R. Sevilla, R., Fernández-Méndez, S., Huerta, A.: NURBS-Enhanced Finite Element Method (NEFEM). Arch. Comput. Methods Eng. **18** (2011)
19. Sevilla, R., Fernández-Méndez, S.: Numerical integration over 2D NURBS-shaped domains with applications to NURBS-enhanced FEM. Finite Elements in Analysis and Design **47**(10), 1209–1220 (2011)
20. Slawski, M.: Non-negative least squares: comparison of algorithms, available at: `https://sites.google.com/site/slawskimartin/code`
21. Sommariva, A.: CUB_RS: MATLAB toolbox for algebraic cubature on NURBS-shaped domains, available at:
`https://www.math.unipd.it/~alvise/software.html`
22. Sommariva, A., Vianello, M.: Compression of multivariate discrete measures and applications. Numer. Funct. Anal. Optim. **36**, 1198–1223 (2015)
23. Sommariva, A., Vianello, M.: Computing Tchakaloff-like cubature rules on spline curvilinear polygons. Dolomites Res. Notes Approx. **14**, 1–11 (2021)
24. Sommariva, A., Vianello, M.: inRS: implementing the indicator function of NURBS-shaped planar domains. Appl. Math. Lett. **130** (2022)

25. Sommariva, A., Vianello, M.: TetraFreeQ: tetrahedra-free quadrature on polyhedral elements, submitted. arXiv:2211.16620 (2022)
26. Tchakaloff, V.: Formules de cubatures mécaniques à coefficients non négatifs (French). Bull. Sci. Math. **81**, 123–134 (1957)
27. Wilhelmsen, D.R.: A Nearest Point Algorithm for Convex Polyhedral Cones and Applications to Positive Linear approximation. Math. Comp. **30**, 48–57 (1976)
28. Wriggers, P., Hudobivnik, B., Aldakheel, F.: NURBS-based geometries: A mapping approach for virtual serendipity elements. Comput. Methods Appl. Mech. Eng. **378** (2021)